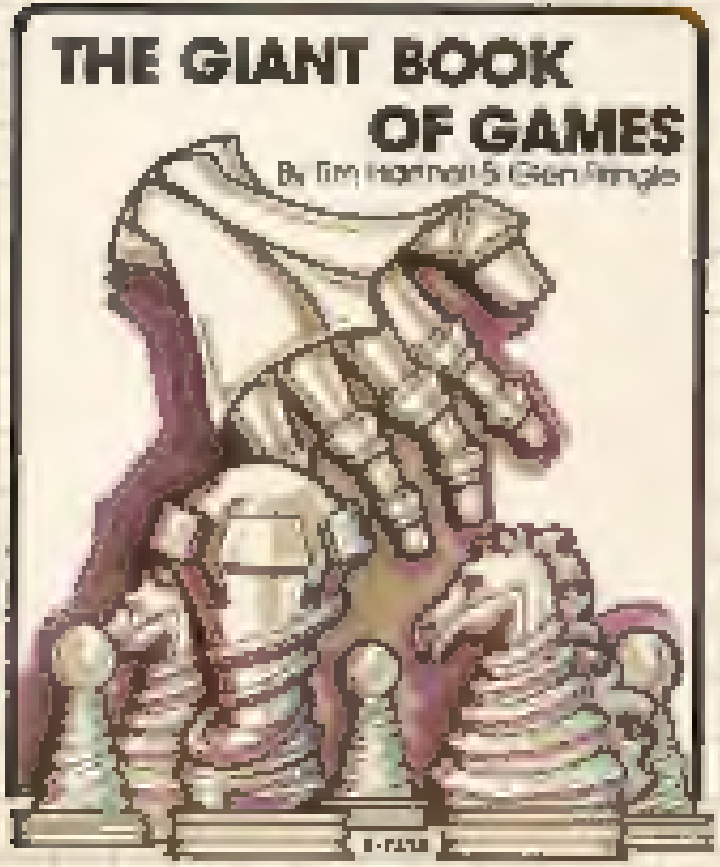
DICK SMITH

VZ200

Personal Colour Computer

THE GIANT BOOK OF GAMES

By Tim Hartnell & Glen Tenglo



First published in Australia by:

Dick Smith Management Pty. Ltd.
Villa Rd,
North Ryde, NSW, 2113

First printing September, 1983

ISBN 0 949772 22 4

Dick Smith Catalogue No. B 7210

Typesetting and artwork by Interface Publications, Level 2, Swain House, 20 Darlinghurst Road, Darlinghurst, Sydney, Australia.

Cover artwork by Geoff Gallop

INDEX

THE GRANT BOOK

OF

VERGO DANSA

The Harlequin

and

Other Prologue

INTRODUCTION

INTRODUCTION

Once upon a time it was not easy to write and play games with computers. Back in the dim, dark past, it could cost you $50 or $100 per hour just for remote access to a time-shared mainframe via a terminal. And the only output you had was on the printout of a noisy, hard to read teleprinter. Think what these restrictions would have done to your current pastime of spending hours lovingly debugging, and improving, your latest masterpiece.

We, fortunately, do not have to suffer such medieval conditions. The luxury of having a computer of your own, driving a TV screen, with unrestricted access to the machine, allows you to work for as long as you like creating games and other programs.

In this book, you'll find a collection of our favorite computer games. We hope some of them will stick by us and running on your computer, and are stocked with your own favorite programs. They're sure to provide you with a lot of fun in the years ahead.

(By the way, from now on, we'll use "S" in this book, rather than "GO" which sounds too much like the "royal we". "S" makes to read better.)

I've deliberately written the book so that it is "bite-sized". That is, in contrast to many other collections of games programs on the market, I've avoided you'll need to enter and improve the programs once you get them up and running. Therefore, with some of them lines included program breakdowns, some line by line, so you know what each section of code is doing. As well, the introduction to each section discusses a bit about the philosophy of writing games of that type, has a word or two to say about the basic algorithms which you can use to create games of that sort yourself, and generally aims to give you a solid background against which you can expand and develop your own game-writing skills.

Near the back of the book you'll even find a section devoted entirely to games ideas which you can convert into computer games.

To maintain the length of program you can get into your machine and to keep program entry time down to a minimum few of the programs include instructions. These are in the text. However, if you have the memory long the inclination you can easily add a "Do you want instructions" line as part of the initialisation subroutine, and then include a short-cut form of the instructions from the text in your program.

I think we should get to the fun parts immediately.

Happy game-playing.

Tim Hartnell
Glen Pringle

Board Games

Board Games

Some games could not have existed without computers.

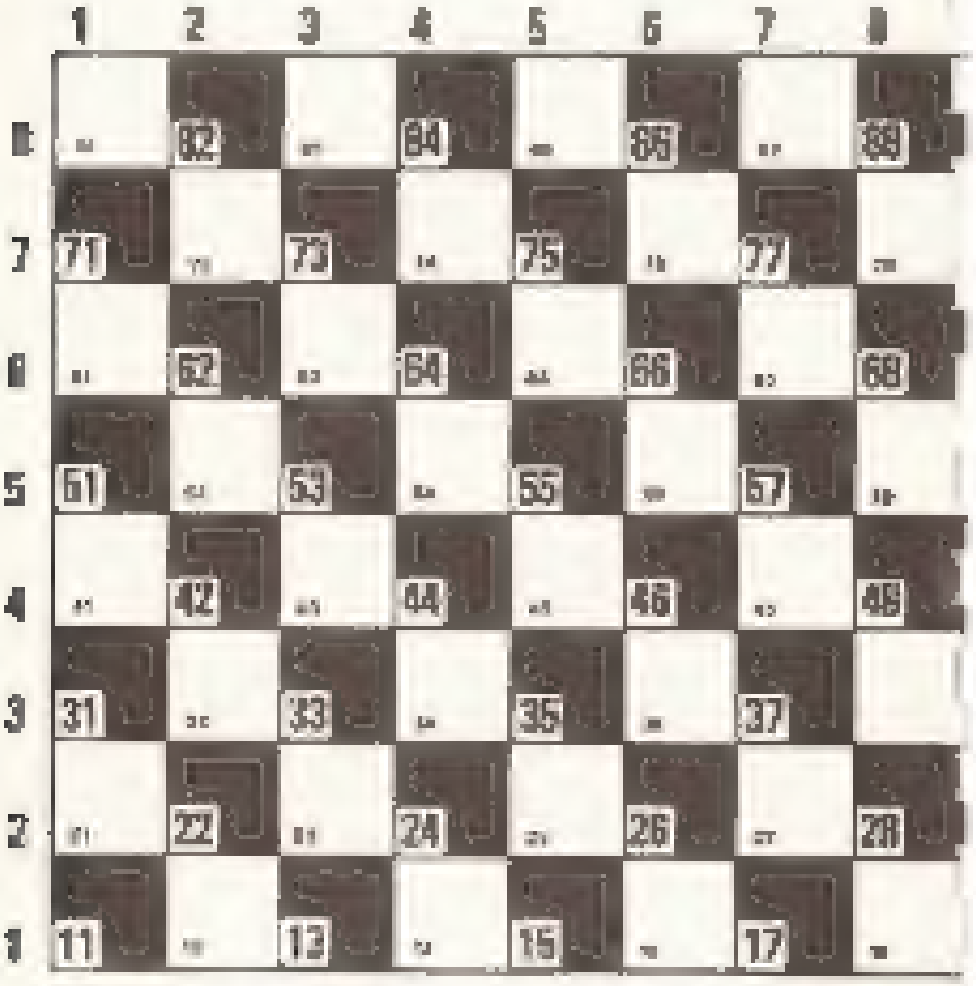
Think of the whole range of Space Invader types of games and the Pacman game. These games have not come into being before the computer - using a TV-like screen for output - was developed. Many other games, of course, predate the computer by hundreds of years. However, more and more of these traditional games have now been computerised, with board games being as popular with computer games players as they were with players in pre-computer days.

In this section of the book, we have a generous crop of board games:

- GEMOKU
- FOUR IN A ROW
- CHECKERS
- KNIGHTSONTOUR

We'll start this section of the book with a discussion on how board games can be easily modified to make it possible for a computer to handle them. There is a common thread which holds together all the programs in this section of the board, a thread which you can also use to help you develop programs to play your favourite board games.

Look first at this diagram. It shows a checkerboard numbered to make it easy for the computer to handle.



You can identify any square on the board by referring to the number along the left hand side, (such as 3), then the number along the top (such as 5). In this case, the line numbered 3 along the left hand side and the line numbered 5 along the top meet at the square numbered 35. If you wish to make a move, you'll do so by entering the number of the square you're moving from (such as 53), then the number you're moving to (such as 44) and the computer can understand exactly what you're doing. There is no need to change the numbers entered by the human player into another set in order to allow the computer to interpret them.

That is the first 'trick' of writing board games so your computer can play them. The second is that a board numbered in this way has another great advantage over one which is simply numbered from one to 64 in order. When you move a piece in any direction on this board — no matter where you start — the difference between the squares is the same.

I'll try to explain in that somewhat cryptic statement. If you move one square up and to the right — like the move of a piece in checkers — you will move from, say, 24 to 35; or from 53 to 64; or from 71 to 82. But notice that no matter where you are on the board, the difference between your starting and ending squares is always 11. If you move diagonally up to the left, you'll move from, say, 24 to 33 (move 9), or 64 to 75 (also move 9) or from 22 to 31 (move 9 again).

This predictability makes it relatively simple to create a board which the computer can handle.

How the computer plays

Imagine the computer has a checkers piece on the square numbered 24. It could be programmed to check each square on the board, and every time it found one of its own pieces, would check if there was a human piece on the square numbered the same as the

(such as 3), then the number along the top (such as 5). In this case, the line numbered 3 along the left hand side and the line numbered 5 along the top meet at the square numbered 35. If you wish to make a move, you'll do so by entering the number of the square you're moving from (such as 53), then the number you're moving to (such as 44) and the computer can understand exactly what you're doing. There is no need to change the numbers entered by the human player into another set in order to allow the computer to interpret them.

That is the first 'trick' of writing board games so your computer can play them. The second is that a board numbered in this way has another great advantage over one which is simply numbered from one to 64 in order. When you move a piece in any direction on this board — no matter where you start — the difference between the squares is the same.

I'll try to explain in that somewhat cryptic statement. If you move one square up and to the right — like the move of a piece in checkers — you will move from, say, 24 to 35; or from 53 to 64; or from 71 to 82. But notice that no matter where you are on the board, the difference between your starting and ending squares is always 11. If you move diagonally up to the left, you'll move from, say, 24 to 33 (move 9), or 64 to 75 (also move 9) or from 22 to 31 (move 9 again).

This predictability makes it relatively simple to create a board which the computer can handle.

How the computer plays

Imagine the computer has a checkers piece on the square numbered 24. It could be programmed to check each square on the board, and every time it found one of its own pieces, would check if there was a human piece on the square numbered the same as the

piece issue is. Is in this example, plus 11 (that is, an 8)? and it would check to see whether the square which was 77 (that is, 40) was clear. If it found that all these conditions were true, the computer would jump over square 33 (the square 40, and capture the piece currently occupying 33.

This, in essence, is how many computer board games -- from checkers, through reversi, to chess -- work, based on a check 8 by 8 grid numbered in this way.

If you were writing chess you could specify the moves of the knight by knowing it can always move to squares which are the following 'numerical distance' from the square on which the piece now sits:

```
-21  -12  -8  -19  -27  -32  -33  -31
```

Try it now, by placing a coin on square number 20, and work it as if it were a knight, working out the mathematical relationship between the starting square, and the square you're moving to. You should find the differences are the same as the numbers listed above.

The programs in this section, and in many other parts of this book, were written in a 'top down' manner, in an attempt to produce a structured program which is relatively easy to follow and modify. Many of the programs were written by entering the following 'bare bones' as the model:

```
10   REM NAME OF PROG
20   GOSUB 8000:REM INITIALISE
30   GOSUB 9000:REM PRINT BOARD
40   GOSUB 1000:REM COMPUTER MOVES
50   GOSUB 9000:REM PRINT BOARD
60   GOSUB 7500:REM HUMAN MOVES
70   CHECK IF HUMAN HAS WON OR COMPUTER HAS
          WON AND IF SO BREAK TO END OF
```



```
          GAME MESSAGE...
80   GOTO 30
```

This structure was written, in hardly all cases, before I had the slightest idea how I was to accomplish the goals I had set myself for each of the subroutines.

All programs were written and completely on paper before the computer was even turned on, so that a version of the game could be 'hand run' before starting to actually program it into the computer. This meant the ideas seemed to be sought right at the beginning.

Among the many great advantages I discovered regarding programming on paper in this way was the willingness with which I completely discarded whole blocks of code if they were found to be unworkable. It is much more difficult to decide to erase a whole section of code from a program once you actually have it in the computer than it is to just take up a piece of paper. The computation -- what lines of program are actually in the computer -- is too fiddle with them, in an attempt to make them work, at least after a fashion.

Working on paper, then, leads to quite feeling code which really should not be in a program something like jigsaw pieces together to make it work. Working with a structured outline, such as that I've described, makes it simple to know which parts of the program carry out which task. I know, for example, in the pelican game that the board is printed by the subroutine starting at line 9000. Therefore, if the board comes out looking a bit odd, I know immediately which section of the code I should concentrate on.

As well as helping in the early 'get it working' stage, a structured program helps in further developments. Once you have, for example, your own checkers

program up and running, you can go back at a later time in the future to try and make it play a little better by going just to that part of the code which secure the computer's game. You will not have to wade through page after page of code, trying to work out just what each line does, and which particular lines control the computer's play.

The methods I've just outlined are used throughout this, and most other sections, of the book. It may prove instructive to look through some of the programs to follow through the listing, and work out which section does what. In many programs, you'll see that I've included lines of asterisks as a new statement. These break the program into separate modules which should help you follow the program through.

GOMOKU

You'll find GOMOKU an easy game to learn, but one which is almost impossible to win. The computer plays extremely well in this program, which is based on one written by Graham Sharples.

You have to try to get five of your pieces (the X's) in a row in any direction, while the computer is trying to do the same.

Here a look at three board positions from the start of one game I played against the program. Studying the principle will show you how the game unfolds, and will show you how to play it:

```
  1 2 3 4 5 6 7 8
1 - - - - - - - - 1
2 - - - - - - - - 2
3 - - - O - - - - 3
4 - - - - - - - - 4
5 - - - - X - - - 5
6 - - - - - - - - 6
7 - - X - - - - - 7
8 - - - - - O - - 8
  1 2 3 4 5 6 7 8
```

```
  1 2 3 4 5 6 7 8
1 - - - - - - - - 1
2 - - - - - - - - 2
3 - - - O - - - - 3
4 - - - - - - - - 4
5 - - - - X - - - 5
6 - - - - O - - - 6
7 - - X - X - - - 7
8 - - - - - O - - 8
  1 2 3 4 5 6 7 8
```

```
  1 2 3 4 5 6 7 8
1 - - - - - - - - 1
2 - - - - - - - - 2
3 - - - O - - - - 3
4 - - - - - - - - 4
5 - - - - X - - - 5
6 - - - - O - - - 6
7 - - X - X - - - 7
8 - - - - - O - - 8
  1 2 3 4 5 6 7 8
```

```
    1 2 3 4 5 6 7 8
1   [illegible]       1
2                     2
3                     3
4                     4
5                     5
6                     6
7                     7
8                     8
    1 2 3 4 5 6 7 8
```

```
    1 2 3 4 5 6 7 8
1   [illegible]       1
2                     2
3                     3
4                     4
5                     5
6                     6
7                     7
8                     8
    1 2 3 4 5 6 7 8
```

```
    1 2 3 4 5 6 7 8
1   [illegible]       1
2                     2
3                     3
4                     4
5                     5
6                     6
7                     7
8                     8
    1 2 3 4 5 6 7 8
```

```
    1 2 3 4 5 6 7 8
1   [illegible]       1
2                     2
3                     3
4                     4
5                     5
6                     6
7                     7
8                     8
    1 2 3 4 5 6 7 8
```

Here's the listing for your very own copy of [illegible]:

```
10 [illegible]
20 GOSUB [illegible]
30 GOSUB [illegible]
40 GOSUB [illegible]
50 GOSUB [illegible]
60 GOSUB [illegible]
70 GOSUB [illegible]
80 IF [illegible] THEN PRINT:PRINT "[illegible]":GOTO [illegible]
90 GOTO [illegible]
100 [illegible]
110 [illegible] THEN RETURN
120 [illegible]
130 PRINT [illegible]
140 PRINT [illegible]
150 FOR [illegible]
```

```
170 FOR [illegible] TO 8
180 [illegible]
190 IF [illegible] THEN PRINT "[illegible]";
200 IF [illegible] THEN PRINT "[illegible]";
210 IF [illegible] THEN PRINT "[illegible]";
220 NEXT [illegible]
230 PRINT [illegible]
240 NEXT [illegible]
250 PRINT TAB([illegible]);"[illegible]"
260 RETURN
270 PRINT:PRINT
280 PRINT [illegible]"PLEASE ENTER YOUR MOVE[illegible]"
290 INPUT [illegible]
300 [illegible]
310 IF [illegible] OR [illegible] THEN [illegible]
320 [illegible]
330 [illegible]
340 RETURN
350 [illegible]
360 [illegible]
370 FOR [illegible] TO [illegible]
380 [illegible]
390 [illegible]
400 IF [illegible] THEN [illegible]
410 NEXT [illegible]
420 IF [illegible] THEN PRINT:PRINT "[illegible]"
430 [illegible]
440 IF [illegible] THEN [illegible]
450 IF [illegible] THEN [illegible]
460 [illegible]
470 FOR [illegible] TO [illegible]
480 [illegible]
490 IF [illegible] THEN [illegible]
500 FOR [illegible] TO [illegible]
510 GOSUB [illegible]
520 [illegible]
530 IF [illegible] THEN [illegible]
540 IF [illegible] THEN [illegible]
550 IF [illegible] AND [illegible] THEN [illegible]
560 [illegible]
570 NEXT [illegible]
580 IF [illegible] THEN [illegible]
```

FOUR IN A ROW

[illegible]

[illegible]

The computer [illegible] [illegible]

[illegible]

[illegible]

[

CHECKERS

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

KNIGHTSBRIDGE

[illegible]

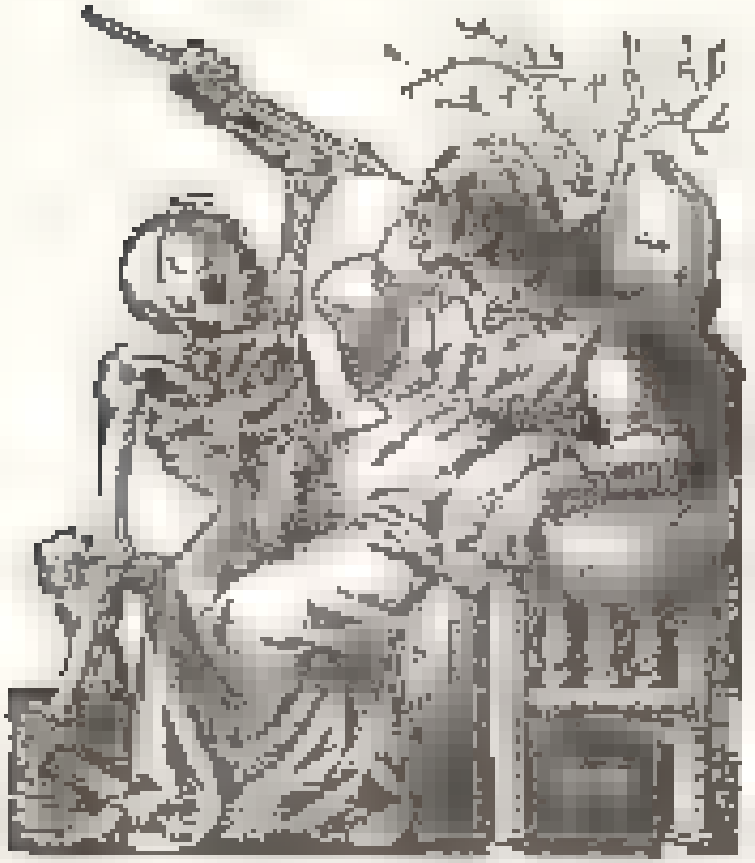
You capture by [illegible]

The [illegible] the consoles [illegible]

[illegible]

[illegible]

Adventure



Adventure

[illegible]

[illegible]

ADVENTURE [illegible] ADVENTURE [illegible]

[illegible] ADVENTURE [illegible]

[illegible] ADVENTURE [illegible]

[illegible] ADVENTURE [illegible]

[illegible] ADVENTURE [illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

STRONGHOLD OF THE DWARVEN LORDS

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

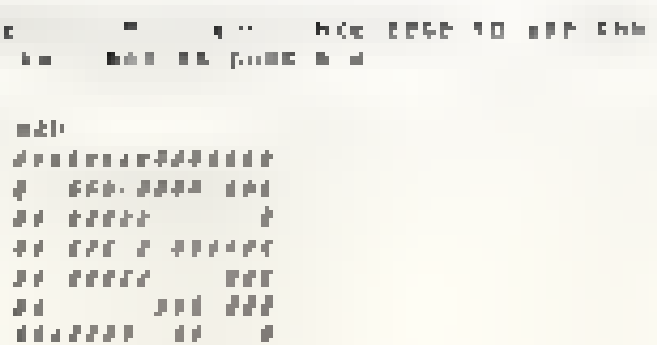
[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

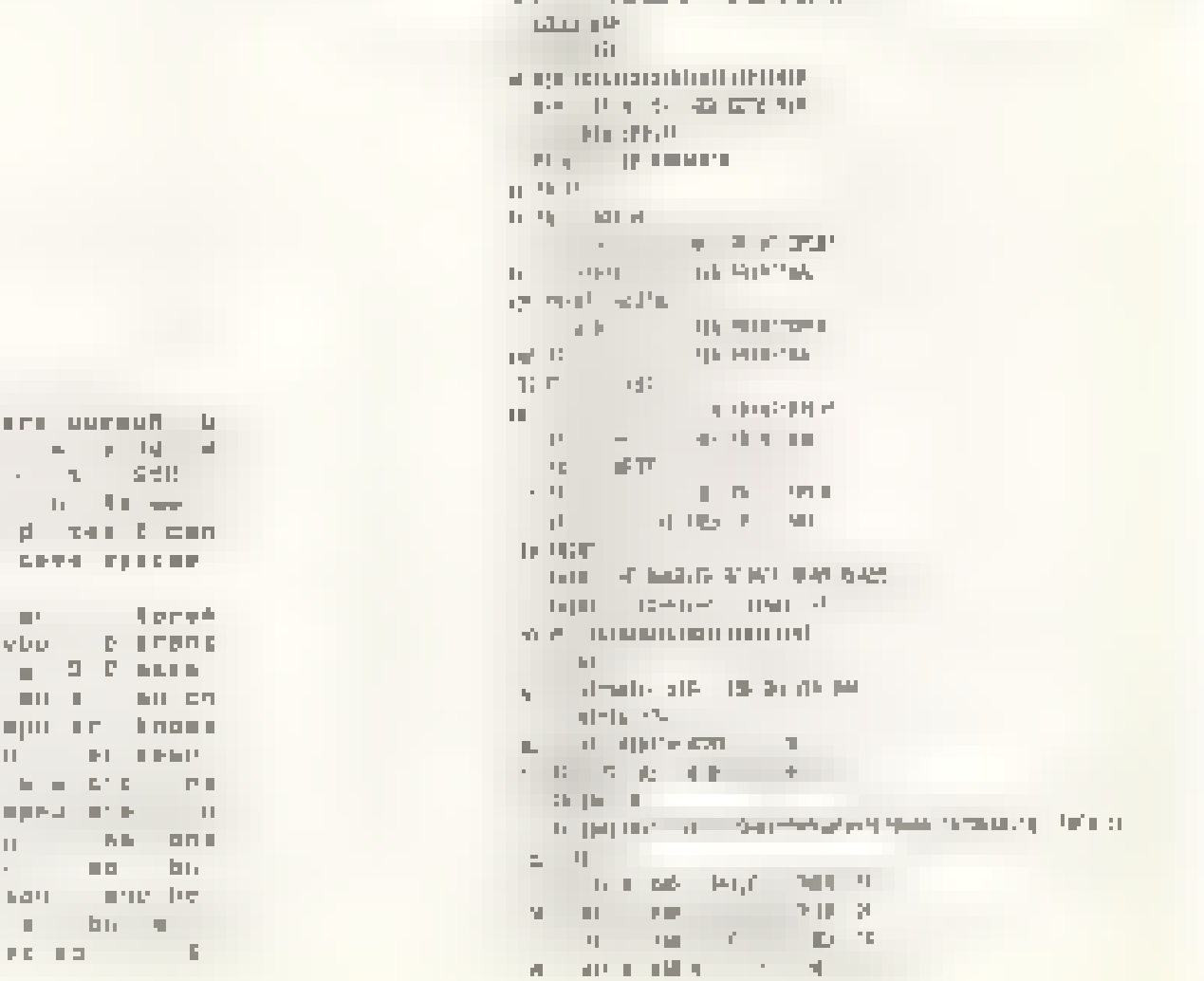


[illegible]

[illegible]

[illegible]

[illegible]



THE DUKE OF DRAGONFEAR

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

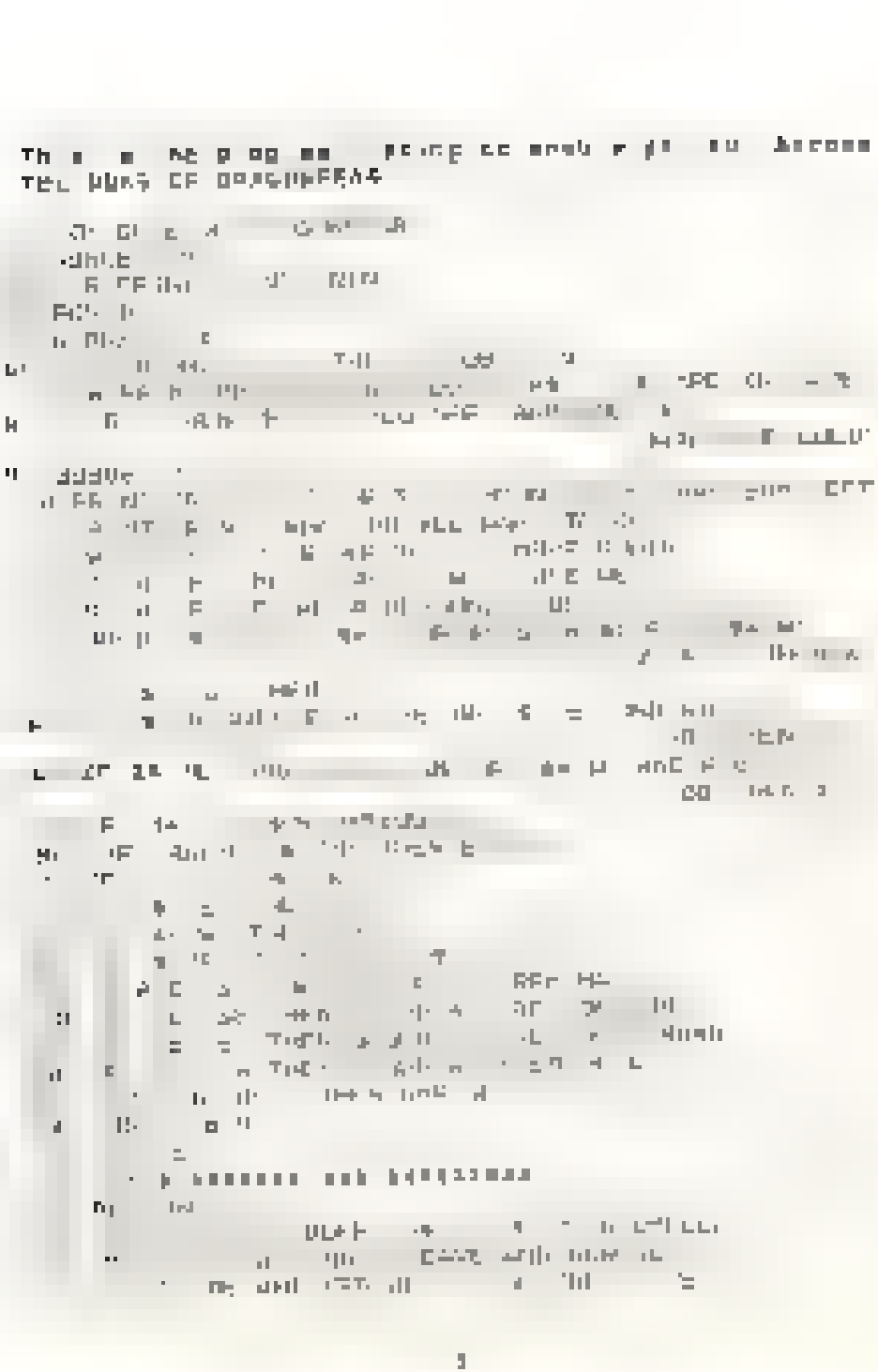
[illegible]

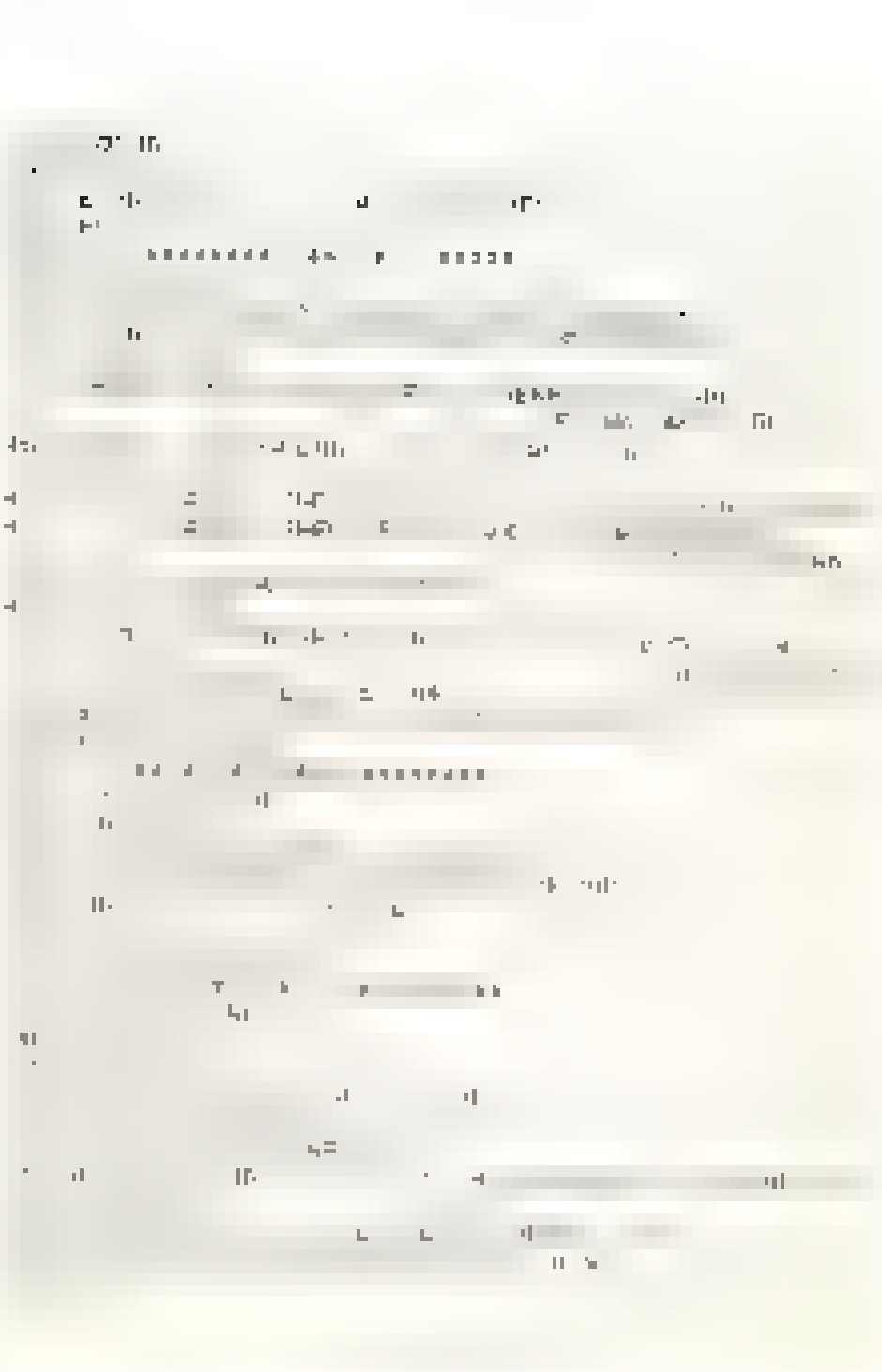
[illegible]

[illegible]

[illegible]

[illegible]





Simulations

Simulations

[illegible]

[illegible]

[illegible]

MISTRESS OF XENOPHOBIA

[illegible]

[illegible]

[illegible]

[illegible]

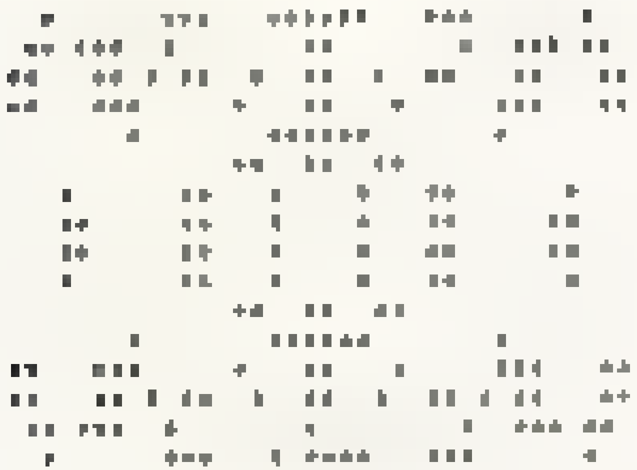
RURAL PURSUITS

[illegible]

[illegible]

[illegible]

[illegible]



CHAIRMAN OF THE BOARD

[illegible] CHAIRMAN OF THE BOARD

[illegible]

[illegible]

```
[illegible]
```

[illegible]

Dice Games

Dice Games

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```
[illegible]
```

```
[illegible]
```

[illegible]

[illegible]

[illegible]

CHEMIN DE COMPUTER

CHEMIN DE COMPUTER is based, [illegible] on Chemin de Fer. [illegible] the computer.

[illegible]

There are [illegible]

As in many other programs in this book, [illegible]

[illegible]

SEVEN/ELEVEN

[illegible]

[illegible]

[illegible]

NO SWEAT

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```
[illegible]
```

ONE-AND-TWENTY

[illegible]

[illegible]

[illegible]

SNAKES EYES

[illegible]

[illegible]

[illegible]

UNDER 'N' OVER

[illegible]

[illegible]

Again [illegible]

[illegible]

[illegible]

[illegible]

[illegible]

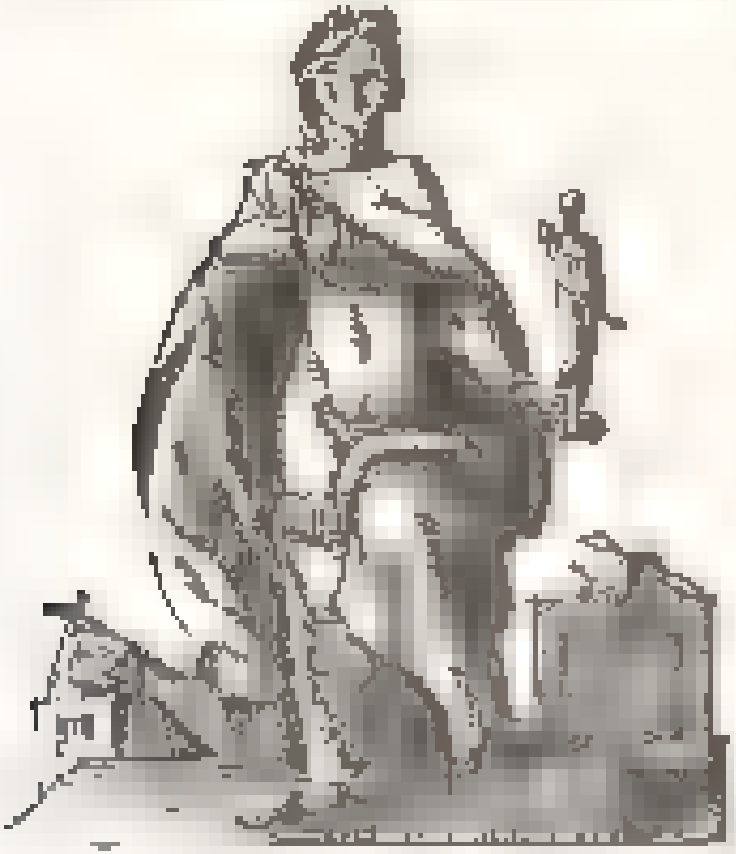
THE OUTCOME

[illegible]

[illegible]

[illegible]

Artificial Intelligence



Artificial Intelligence

[illegible]

[illegible]

[illegible]

ELECTRONIC BRAIN

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

AMANUENSIS

[illegible]

[illegible]

[illegible]

Just for Fun

Just for Fun

[illegible]

[illegible]

[illegible]

[illegible]

FASTER

[illegible]

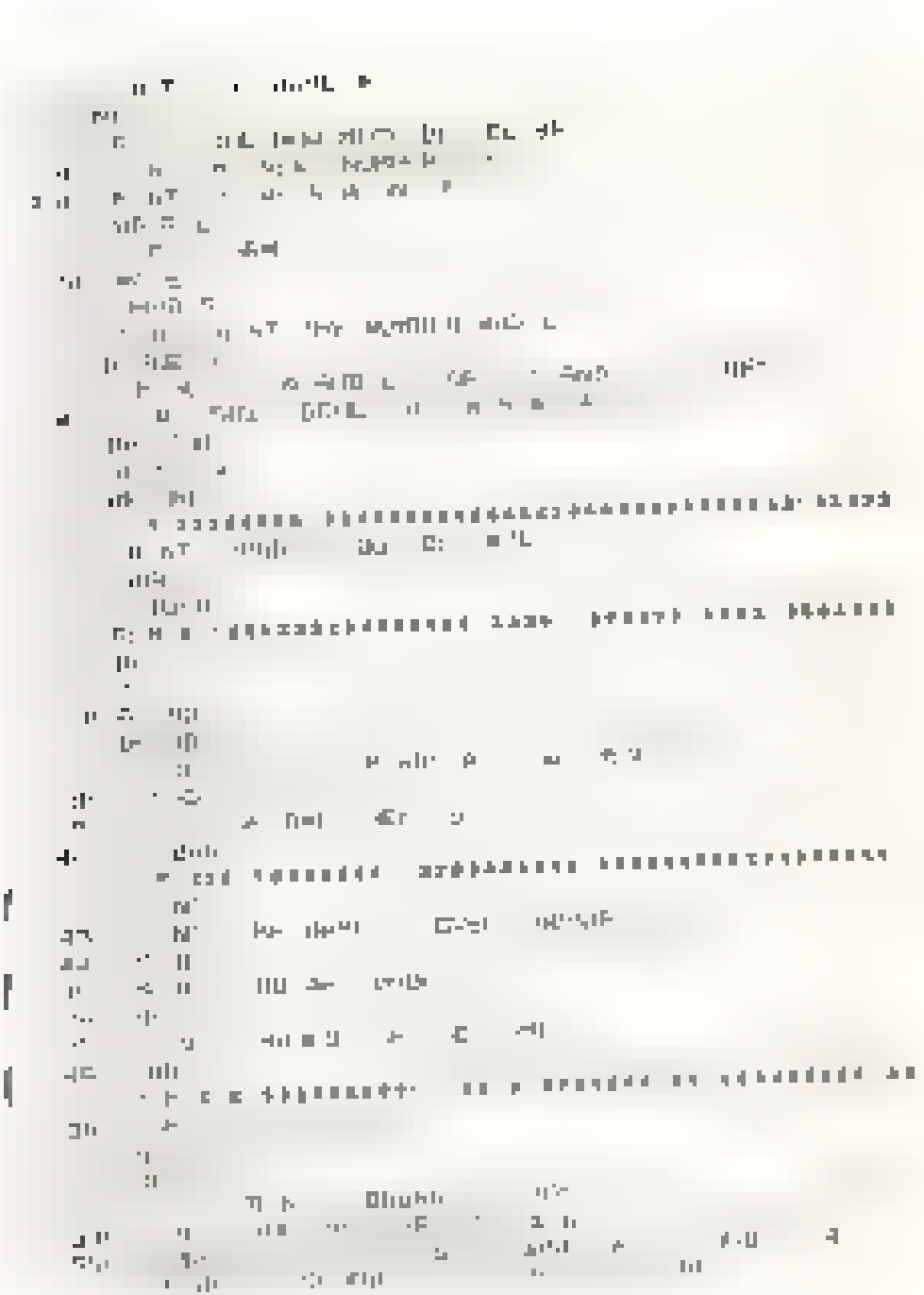
INNER SPRING

[illegible]

```
[illegible]

     [illegible]
     [illegible]

     [illegible]

     [illegible]
     [illegible]
     [illegible]

     [illegible]
     [illegible]
     [illegible]
```

[illegible]

```
[illegible]
```



ROBOT MINEFIELD

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

Fun with your Printer
CELESTIA

[illegible]

The [illegible] famous computer [illegible] Conway [illegible] Cambridge [illegible] the UK [illegible] Conway [illegible] Spectrum [illegible]

[illegible]

The [illegible]

[illegible]

[illegible]

[illegible]

Don't worry [illegible]

[illegible]

[illegible]

[illegible] one [illegible] thing to give [illegible]
[illegible] computer

```
[illegible]
PRINT "WHEN YOU ARE
READY TO BEGIN"
[illegible]
PRINT [illegible]
[illegible]
PRINT [illegible] PRINT
[illegible]
```

BILLBOARD

[illegible]

[illegible]

[illegible]

Space Games

Space Games

[illegible]

MOONLANDER

[illegible]

[illegible]

```
[illegible]
```

HYPERWAR

Brain Games

Brain Games

[illegible]

[illegible]

[illegible]

[illegible]

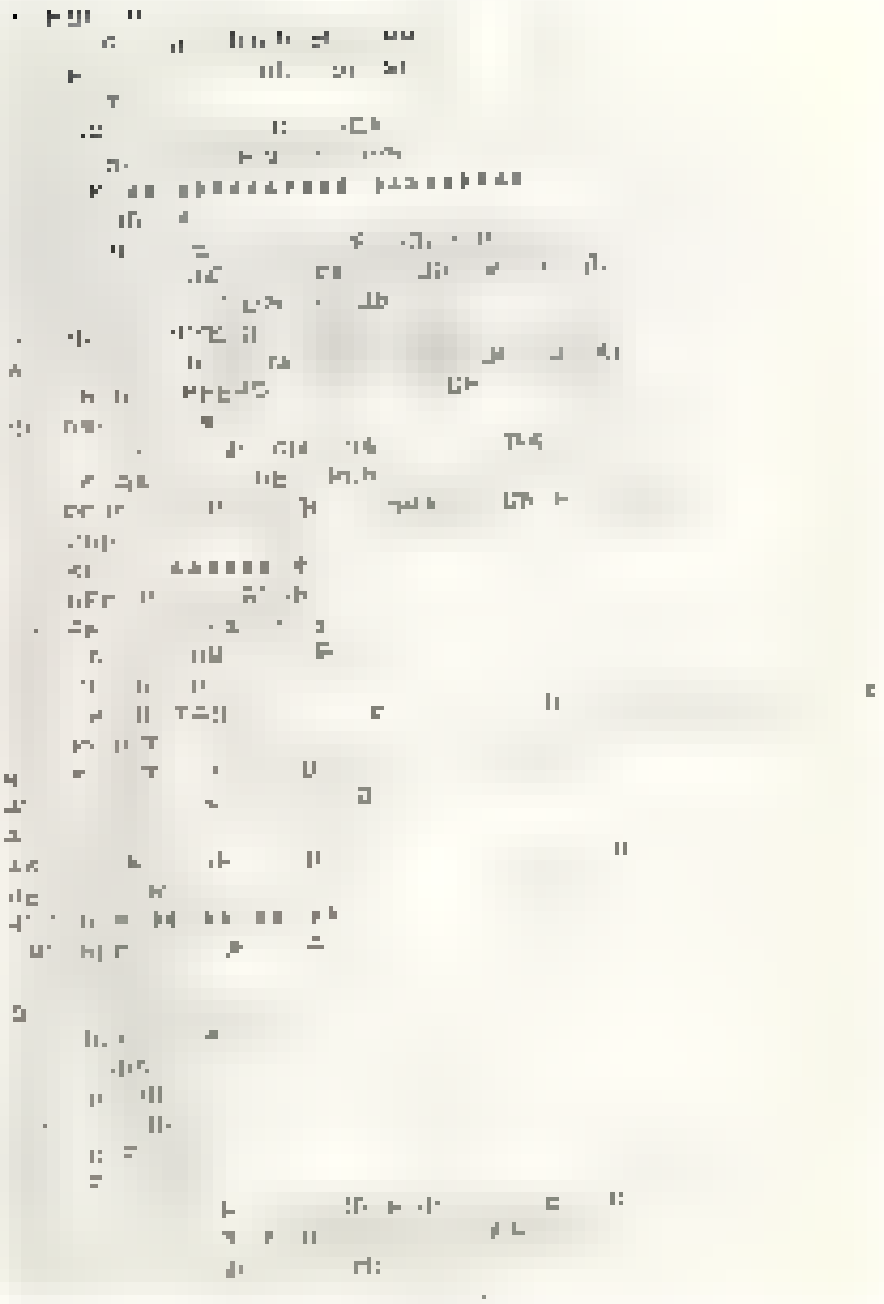
[illegible]

IDAHO STARS

[illegible]

[illegible]

[illegible]



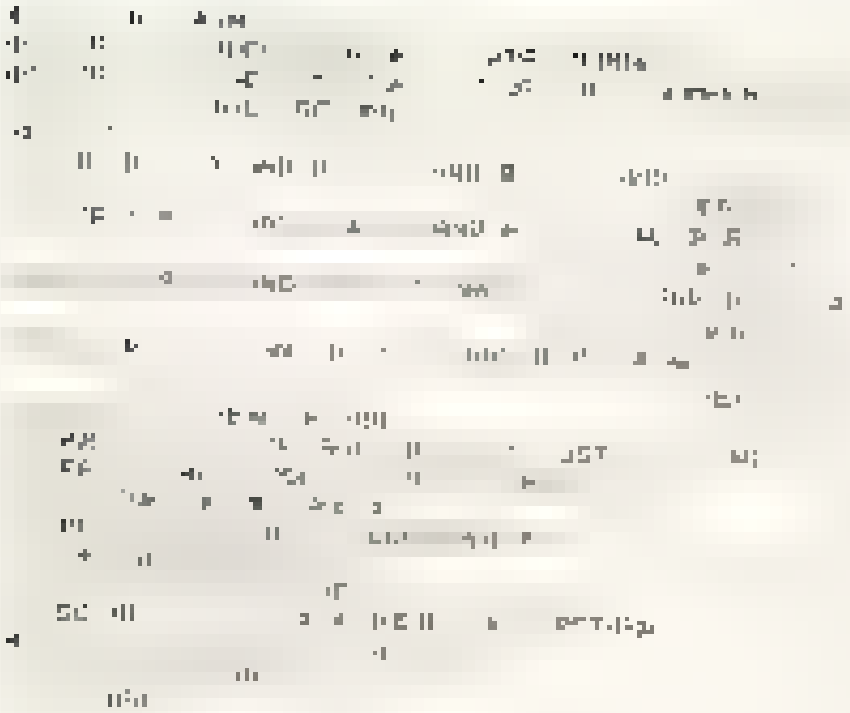
EXECUTIONER

CUBIC

[illegible]

[illegible]

[illegible]

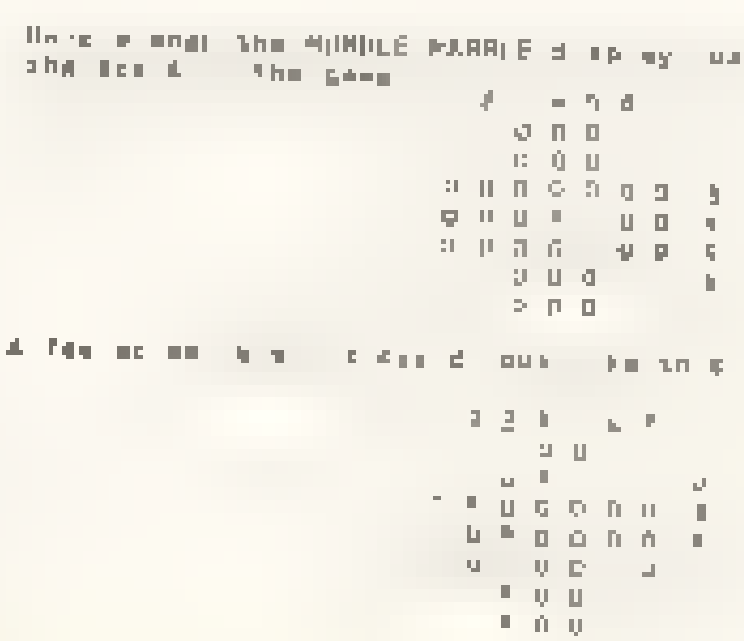


MUMBLE MARBLE

[illegible]

[illegible]

[illegible]

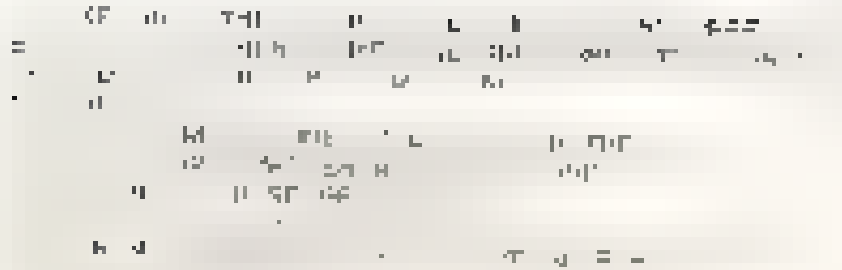


CYCLOTRON

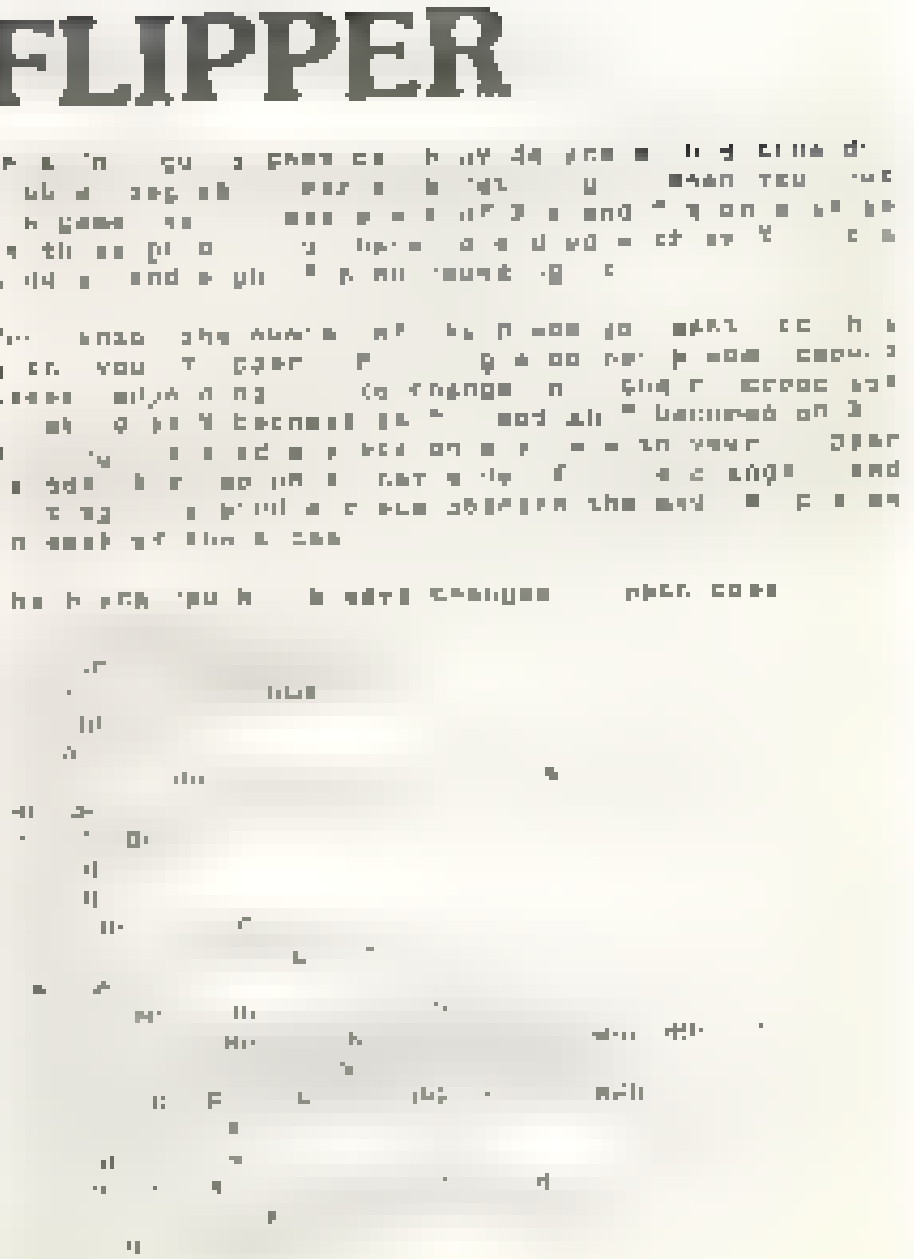
[illegible]

[illegible]

[illegible]



FLIPPER



Creating your own Games

Creating Games

[illegible]

GAMES TO ADAPT

[illegible]

[illegible]

[illegible]

[illegible]

ZENDO: This game is also similar to FOUR ACE however, you aim to get rid of all your chips, rather than win them. The player of the game is paradoxically, the person who has lost the most rounds. You need three dice for this game, and a supply of chips which are distributed equally between the two players. A bet is made, and perhaps, before the players case to be turns to throw the three dice. The throws are scored as follows, with 'winning throws' coming first:

2 1 1; 2 2 1; 4 2 1; 3 6 2; 5 2 5; 4 4 4;

2 2 4; 6 2 2; 3 1 1; 5 3 6; 4 5 6; 3 3 4;

5 3 6; 3 2 1; 1 1 6; 3 1 2; 5 1 4; 5 5 3;

2 2 1; 4 2 2; 2 4 3; 6 5 5

If the combination thrown is not in the list, the mise the money, with the highest total winning.

BARBOOTH: Also known as BARBOOTH, this game is played with two dice, and features winning and losing scores. It is suggested you start with a pool of 50 chips, and give both players a starting stake of 25 chips each, charging two chips per player per round. The players take it in turns to throw the dice. Winning throws are 2 3; 6 6; 6 3; and 5 5. Losing scores are 1 1; 3 2; 2 1 and 4 4. If the players both get winning scores, the one closest to the start of the list wins round 1, 5 5 is better than 6 6. If both throw the same winning score, the dice gets the money. If the player throws a combination which is not a winning one nor a losing one, and the other throws a losing throw, the loser's chips go into the pool, and the other player regains his 50 chips.

If one player throws a winning throw, and the other a non-winning combination, the winner gets his own chips back, as well as those bet by the loser. If, however, one player throws a winning combination, and the other a losing combination, the two can withdraw the same number of chips as each has bet to carry. The game continues until either player, or the bank, runs out of chips. The player reaching the maximum number of chips is the point wins. Note that this way of distributing the chips and scores is not the same as BARBOOTH when played as a 'non-mathematical' gambling game, but in a pay-off scheme which is relatively easy to computerise, and makes for an interesting game.

FIVE THOUSAND: There are no prizes for guessing that the first player to score over 5000 points in this game is the winner. Played with as many players as you like, all you'll need to the way of game equipment is five six-sided dice. Each player takes it in turn to throw the five dice, and scoring is carried out in the following way:

- a 5 is worth 100
- a 1 is worth 50
- three 5's thrown in one go are worth 500
- three 1's thrown in a go score 300
- three of a kind (apart from 1's and 5's, are worth 100 times their value (that is three 6's are worth 600 points. and three 2's are worth 200
- if the sequence 1, 2, 3, 4 and 5 is thrown in a single go, this scores 1500
- 2, 3, 4, 5 and 6 in one throw are worth 1500 points

Note that 4's and 6's do not score in this game, except as part of a sequence. Once you've thrown, you can — if you like — throw the non-scoring dice again. However, if the next throw gains no score, the turn ends, and the overall score for the throw is zero. As long as a scoring throw has been made, any dice 'left over' can be thrown.

This game rewards the bold and careful player with a reasonable score, but gives him or her no chance against a player who decides to take a chance and try to get a massive score in a single throw.

CARD GAMES: Once you've worked out a routine to get the computer held, shuffle and deal the cards back each a routine is to my version of Concentration in this book! you have the raw bones of a game of pelmanism. Any book of card games will give you more ideas that you can possibly cope with. Here are a few to start you off:

TRENTE ET QUARANTE: This game, which originated in 17th century Europe, is also known as ROUGE ET NOIR. Six packs of cards are used in the routine version, but it works well with a single pack. The cards are dealt into two rows, with the total of each row being examined after each row has had an additional card. Count cards (jack, queen, king) count as ten with all other cards (including the ace) counting as their face value. The moment the total of a row equals or exceeds 31, no more cards are added to that row. However, the dealer keeps adding to the other row until it, too, equals or exceeds 31. The first row is called 'black', regardless of the suits of the cards involved, and the second row is 'red'.

You can bet that, at the end, the red row or the black row total will be closer to 31. You can also bet that the colour of the flag need not be dealt will match the colour name of the winning row. All bets are made against the bank, and winners receive their bet back, plus that amount again inside of and so on. If both rows equal 31, the bank takes half the stake and returns the rest to the player.

CARD GRAPE: This game was designed to get around laws passed by some US States which outlawed crude dice with dice. You use a special deck of 52 cards, made up from the aces, 2's, 3's, 4's, 5's and 6's, from four decks of cards. The game follows the standard rules of craps (which are similar to those used in the game SEVEN/ELEVEN in this book).

In its simplified form, without the somewhat complex betting combinations, you play as follows:

- a total of 2, 3 or 12 is 'craps' and the dealer loses, passing the cards to the next player to deal
- a total of 7 or 11 is a 'natural' and the dealer wins
- any other number (4, 5, 6, 8, 9 or 10) is the player's 'point' and he or she must continue dealing to point until the point is dealt again, at which point the dealer wins. However if a 7 is dealt before the point is achieved, the dealer loses

TOUCH CARD BRAG: This game was the forerunner of poker, and as its name suggests, it is played with hands of three, rather than five, cards. Here are the winning combinations, ranked from highest to lowest:

- 'prial', three of a kind
- 'of a kind': three is possible from the same suit
- 'run', three cards in sequence
- 'flush', three cards from the same suit
- 'pair', two of a kind
- 'high card'

A normal deck is used, and aces rank high, except for low a hand' and 'run', when 2, 3, are beaten only, king, queen. You can vary the game by making it SEVEN CARD BRAG, in which the players split their seven cards into two three-card hands of their choice, or NINE CARD BRAG, when three cards are created.

ACE-DEUCE-JACK: This is a simple game, heavily loaded in the dealer's favour. In which players bet on the likelihood that the next card which appears will be an ace, a two (a 'deuce') or a jack.

PUT AND TAKE: This gambling game uses an eight-sided die, which is spun. The game begins with all players have 10 chips, with 00 in the pot. The players take it in turn to spin the die, and — obeying the instructions which come up on the top — put chips into the pot, or take them out. A player is out when he or she runs out of chips, and the game ends when the pot is empty. The instructions written on the sides of the top are:

- PUT ONE
- TAKE ONE
- PUT THREE
- TAKE THREE
- PUT FOUR
- TAKE FOUR
- PUT ALL
- TAKE ALL

ROULE: This is a simplified form of roulette which appears in the 18th century. It is often found in European casinos. The wheel is divided into 10 segments, numbered from one to nine...below. You can bet on the winning number, whether it will be a red or a black number, even or odd, high or low. Payouts for the first on the wheel, which are usually yellow, the red and black alternate, so that there is, for example, a red six and a black six, a red nine and a black nine, and so on. A (high) number is above five, while 'low' numbers are below it. If you bet successfully on a single number, you'll be paid as many to one, with a pay-to-one return on the 'characteristic' of a number (each of it being 'odd' or 'even').

BOOKS TO READ

As you've probably gathered by reading through this book, I make a point of reading any book on games I may get my hand on. It doesn't matter whether the book is about computer games or the old-fashioned 'play against another human' type. So long as it is a book of games, I want to see it.

Games books are supposed to inspire. Read about a game and you've got the seed of a program. And it doesn't matter if the program doesn't end up playing precisely the game you read about. Many arcade computer games started their lives as an attempt to computerise a more traditional game, and someone got side-tracked.

I hope some of the following lists of books prove as enjoyable to you as they have to me.

*DISCOVERING OLD BOARD GAMES - R C Bell (Shire Publications, Aylesbury, UK, 1980)

*RediscoveringDICE AND DOMINOES - R C Bell (Shire Publications, Aylesbury, UK, 1980)

*DICE GAMES NEW AND OLD - William E Tredd (The Glasgow Press, London and New York, 1981)

*GAME PLAYING WITH BASIC - Donald D Spencer (Hayden Book Co., Inc., Rochelle Park, New Jersey, 1977)

*GAMES ANCIENT AND ORIENTAL AND HOW TO PLAY THEM - Edward Falkener (Dover Publications, Inc., New York, 1961 - a reprint of an original work published by Longmans, Green and Co. in 1892)

*BASIC COMPUTER GAMES - Ed. David H Ahl (Creative Computing Press, Morristown, New Jersey, 1978)

*MORE BASIC COMPUTER GAMES - Ed. David H Ahl (Creative Computing Press, Morristown, New Jersey, 1978)

*20 BASIC PROGRAMS FOR YOUR MICRO - Derrick Daines (Business Technical Books, Butterworths & Co., London and Sydney, 1982)

*TABLE GAMES OF GEORGIAN AND VICTORIAN DAYS - F H D Strutt (Frederick Muller, Reprint, Herts., UK, 1971)

*PET FUN AND GAMES - Ron Jeffries and Steve Fisher (Osborne/McGraw-Hill, Berkeley, California, 1982)

*COMPUTER GAMES FOR BUSINESSES, SCHOOLS AND HOMES - J Vincent Nehlsen and William E Hodges (Winthrop Publishers, Inc., Cambridge, Massachusetts, 1978)

*GAMES YOU MAKE AND PLAY - Pia Heine, Matt Lorimer and Alan Williams (Macdonald and Jane's, London 1978)

*WHAT TO DO AFTER YOU HIT RETURN - Peoples's Computer Company (Hayden Book Co., Inc., Rochelle Park, New Jersey, 1980)

*EVERYMAN'S INDOOR GAMES - Dylan Greenwood (Dent Everyman's Library, London, Melbourne and Toronto 1982)

*A PLAYER'S GUIDE TO TABLE GAMES - John Jacobs (Stackpole Books, Harrisburg, Pennsylvania, 1975)

*GAMES FOR TWO - John Wesley (Prentice, London and New York, 1981)

*THE ADDISON-WESLEY BOOK OF APPLE COMPUTER SOFTWARE 1983 - Jeffrey Stanton and John Dickey, editors (The Book Company, Lawndale, California, 1983)

*PUZZLES WITH DIAGRAMS - Yan Kislagorodsk (Routledge & Kegan Paul, London, Melbourne and Henley, 1982)

*FANTASY ROLE PLAYING GAMES - J Eric Holmes (Arrow Books Inc., New York, 1981)

Here's a rash of games programs, just waiting to be loaded into your Dick Smith V2200 Personal Unique Computer. No matter what kind of games you're interested in, you'll find them here, to entertain you in the home and monitor around. Now you can take full advantage of the colour, sound and memory available on the V2200 to build yourself a library of worthwhile and challenging games programs.

Tim Marshall came to London from Melbourne in 1977 to work on the Australian Express newspaper. While in London, he wrote his first computer book, which proved immediately successful. He has now written more than 30 books on microcomputers, and brings his experience gained from the books together in this wickedly geeky book for your V2200.

Tim was joined in the writing of this book by Giles Pringle, a schoolboy from Lilydale, Victoria who completed his major contributions to the work a week before his fourteenth birthday. Giles intends to make his career in computing, probably as a programmer. As well as being interested in computers, Giles reads a lot (mostly war books) and is interested in international cricket.

You'll find much to entertain you in this volume, including board games such as COMMAND AND CONQUER, as well, there are two adventure programs: SWORDLORD OF THE DRAGON LORDS and THE DUKE OF DRAGONFCAN. You can take part in simulation programs like MASTERS OF MEGAPOLIS or run a company in CHAIRMAN OF THE BOARD. There is a generous selection of dice games, such as ONE AND TWENTY and SHEMIN DE COMPUTER.

If you're looking for new entertainment, jump right into ROBOT MINEFIELD and DANGER PRISON. Out in space, your V2200 is waiting to entertain you with MOONLANDER and HYPERWAR.

This is just a sample of the action waiting for you in this great book by Giles and Tim. Come on in, the games are fine!